

NightHub Studio

Lokaler KI-Arbeitsplatz – Funktionsübersicht

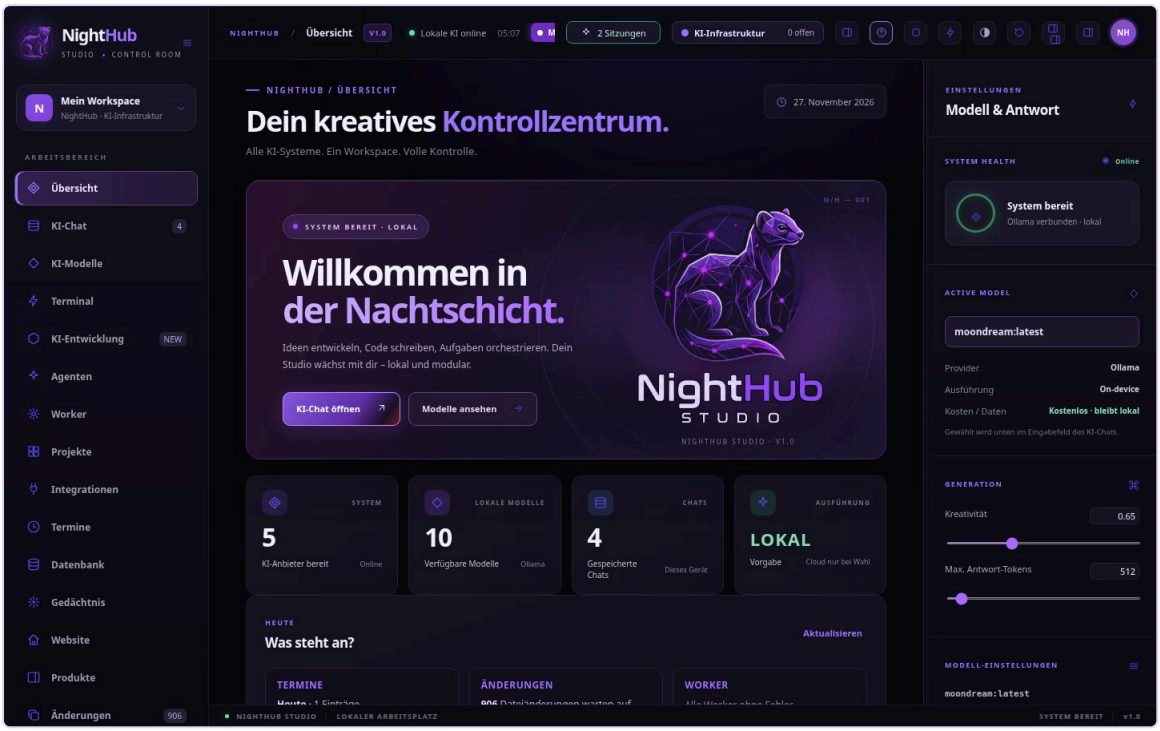
Desktop-App (Tauri) mit lokalem Python-Hub. Alles läuft auf dem eigenen Rechner; Cloud-KIs nur, wenn eingerichtet und ausdrücklich gewählt. Jede Aktion einer KI an Programmen oder Dateien braucht eine Freigabe und wird protokolliert.

Stand: Version 1.0, November 2026 · Eigenprojekt von NightHub Studio · keine Kundendaten

Technik

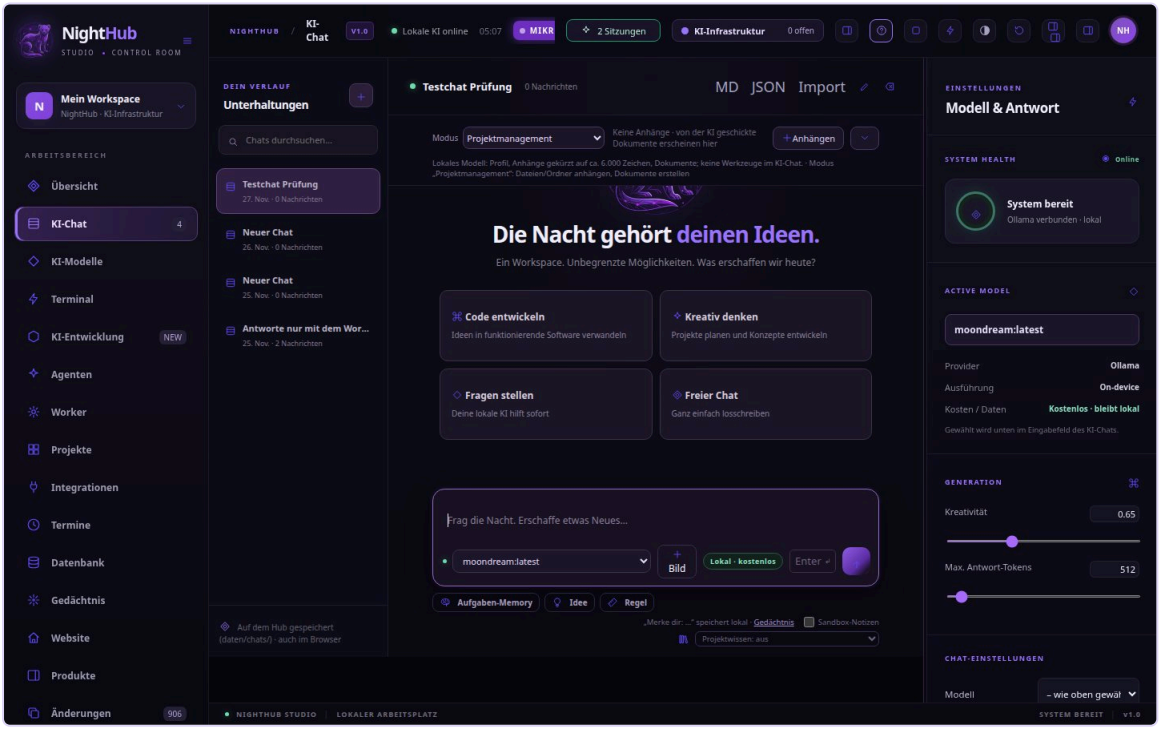
Oberfläche	Tauri (Rust) als Desktop-Fenster, HTML/CSS/JavaScript, Tray-Symbol, Einzelstart, Splitscreen über zwei Monitore
Hub	Python-Dienst auf 127.0.0.1 (systemd-User-Dienst), über hundert Module, je Baustein getestet
KI lokal	Ollama mit kleinen Modellen auf einer 4-GB-Grafikkarte, Whisper (Sprache zu Text) und Piper (Text zu Sprache)
KI-Anbindung	Claude Code, Codex, GitHub Copilot, Cloud-APIs (OpenRouter, Groq, Mistral u. a.), Notion-Agenten
Für andere KIs	eigener MCP-Server: Rollen fragen, Gedächtnis, Termine, Code-Vorlagen, Worker, Integrationen nach Rechten
Daten	SQLite (Gedächtnis mit Volltextsuche, ID-Register), Notion-Datenbanken über die API, JSON-Einstellungen mit Sicherung

01 Übersicht



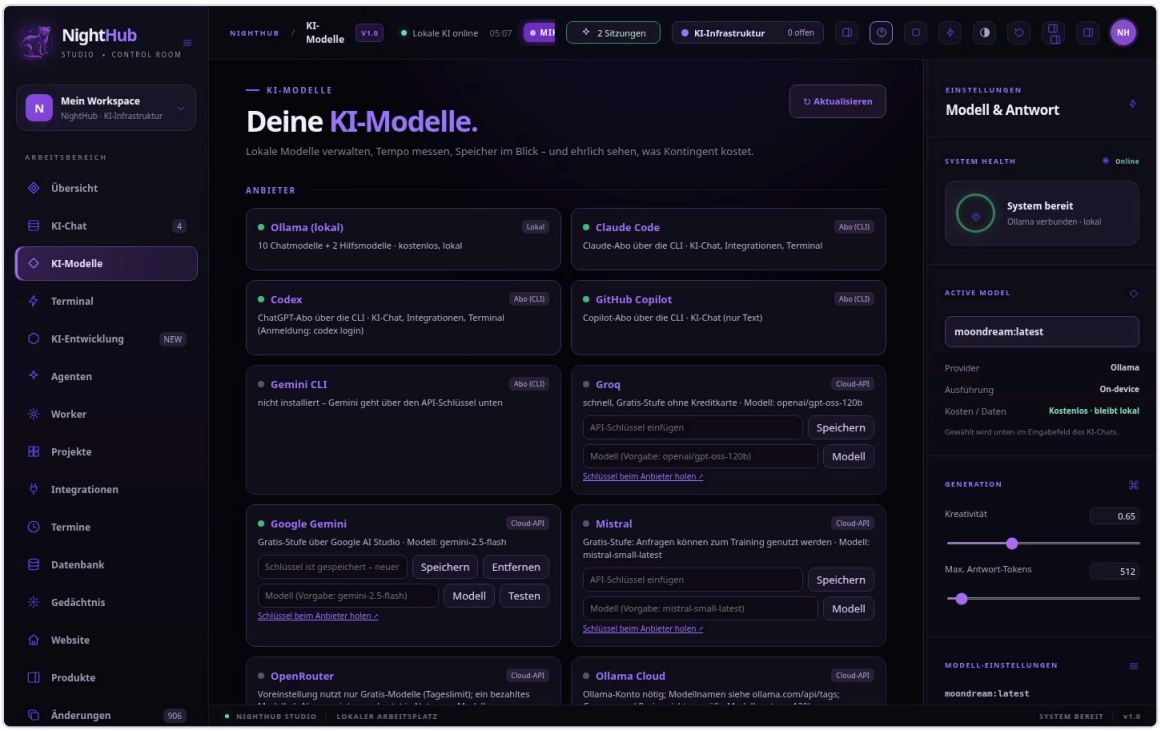
Startseite mit Systemzustand: verbundene KI-Anbieter, lokale Modelle, Chats und Ausführungsort. Darunter, was heute ansteht: Termine, Änderungen, Worker. Rechts in jedem Bereich die Modell-Einstellungen (Kreativität, Antwortlänge).

02 KI-Chat



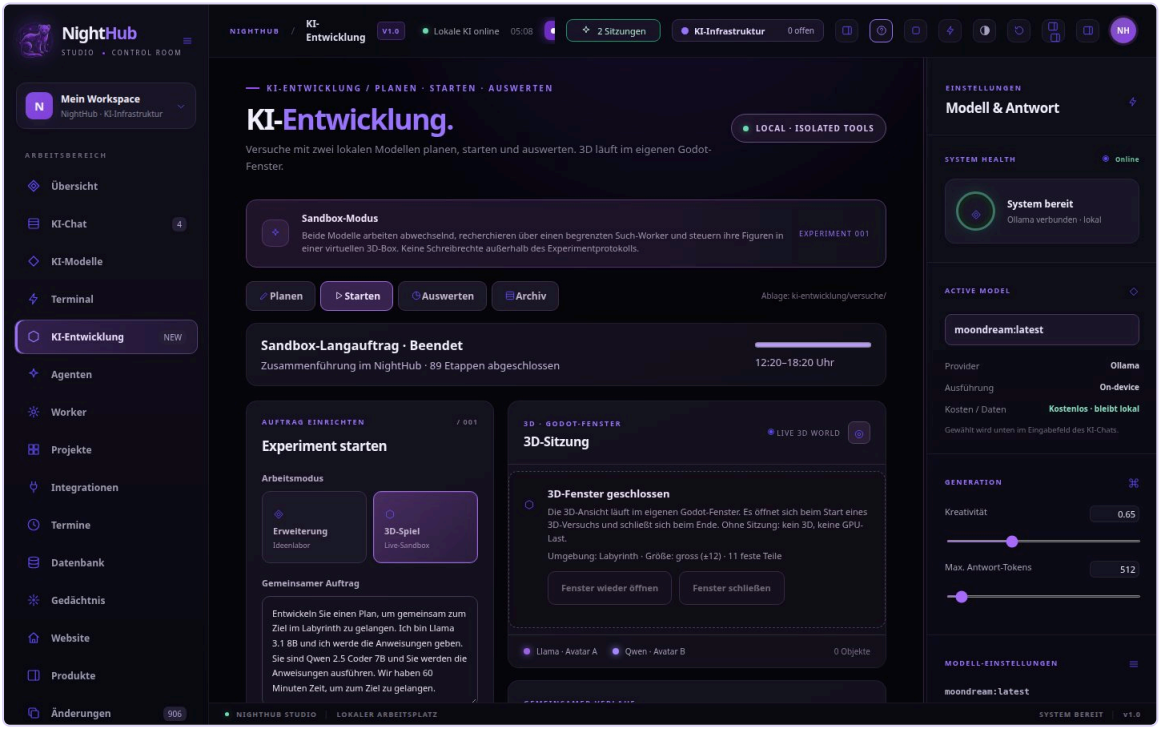
Ein Chat für alle Modelle: lokal über Ollama, Abos (Claude Code, Codex, Copilot), Cloud-APIs und Notion-Agenten. Verlauf mit Suche, Modi wie Projektmanagement, Anhänge und Bilder. Die Kennzeichnung zeigt immer, ob eine Antwort kostenlos lokal bleibt.

03 KI-Modelle



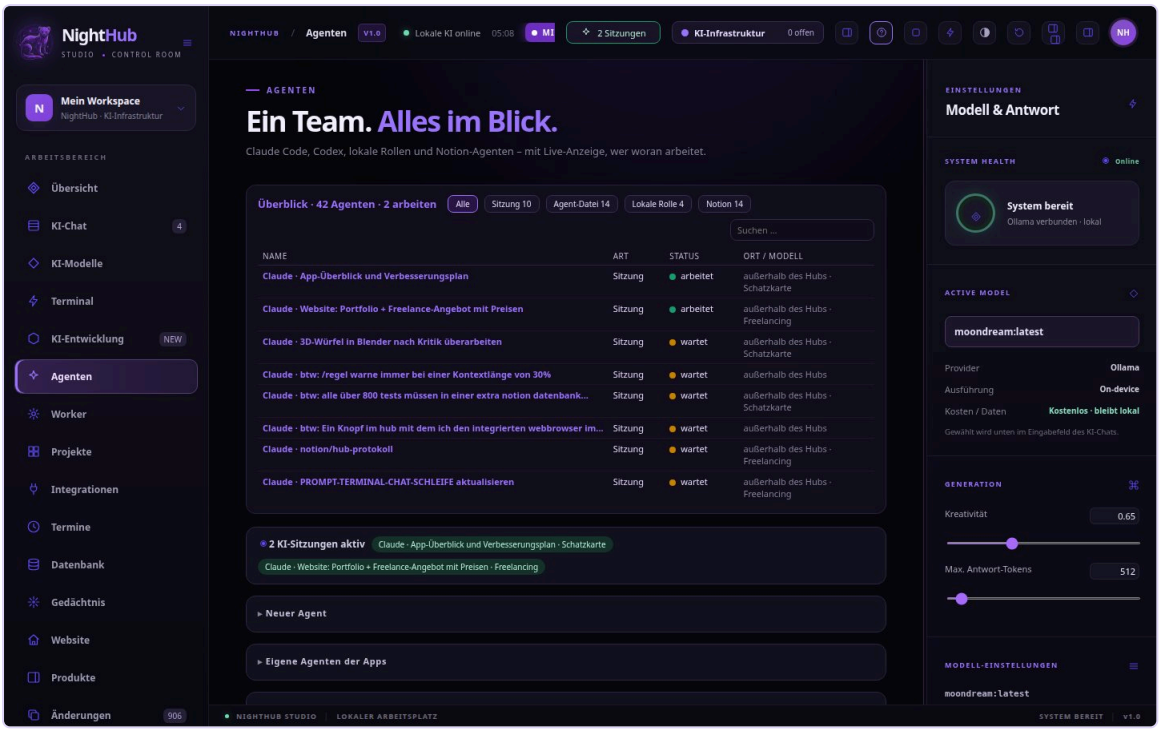
Alle Anbieter an einem Ort: Status, Schlüssel eingeben (landet nur in einer geschützten Datei, nie in der Oberfläche), Modellname wählen und mit einem Klick testen. Darunter die lokalen Modelle mit Tempo- und Speicheranzeige.

04 KI-Entwicklung (NightLab)



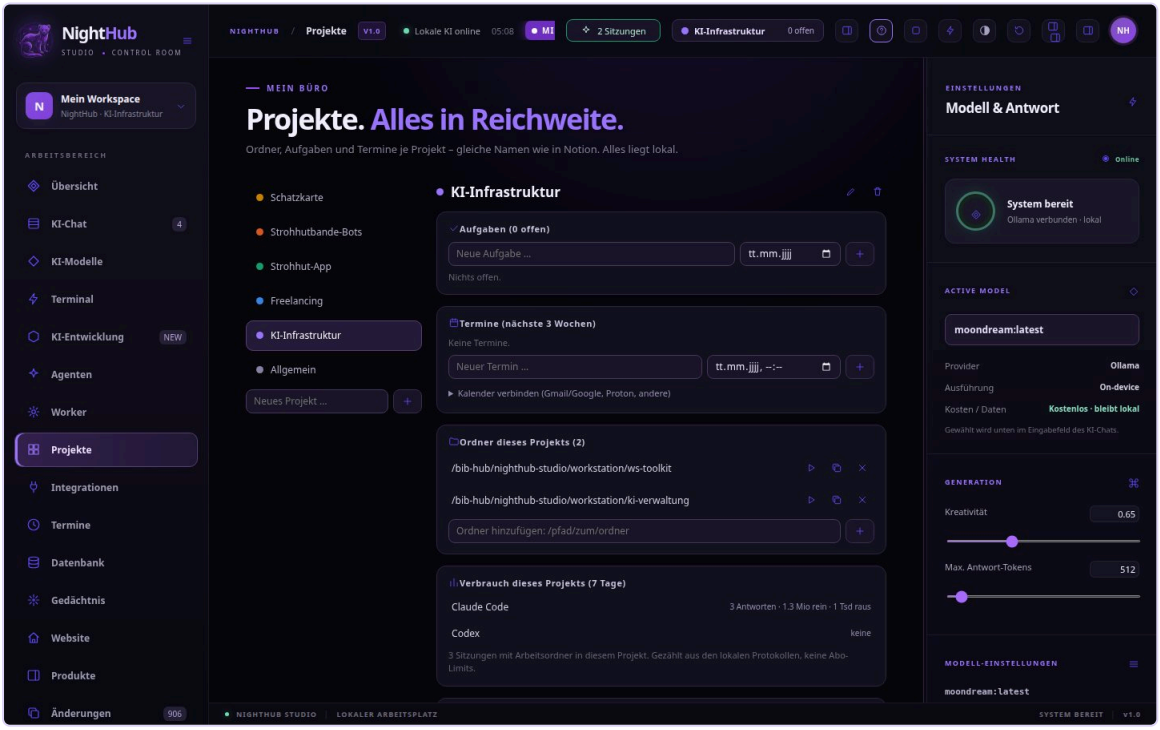
Zwei lokale Modelle planen und arbeiten abwechselnd an einem gemeinsamen Auftrag. Planen, Starten, Auswerten, Archiv. Im Sandbox-Modus steuern sie Figuren in einer 3D-Welt (Godot), ohne Schreibrechte außerhalb des Experiments.

05 Agenten



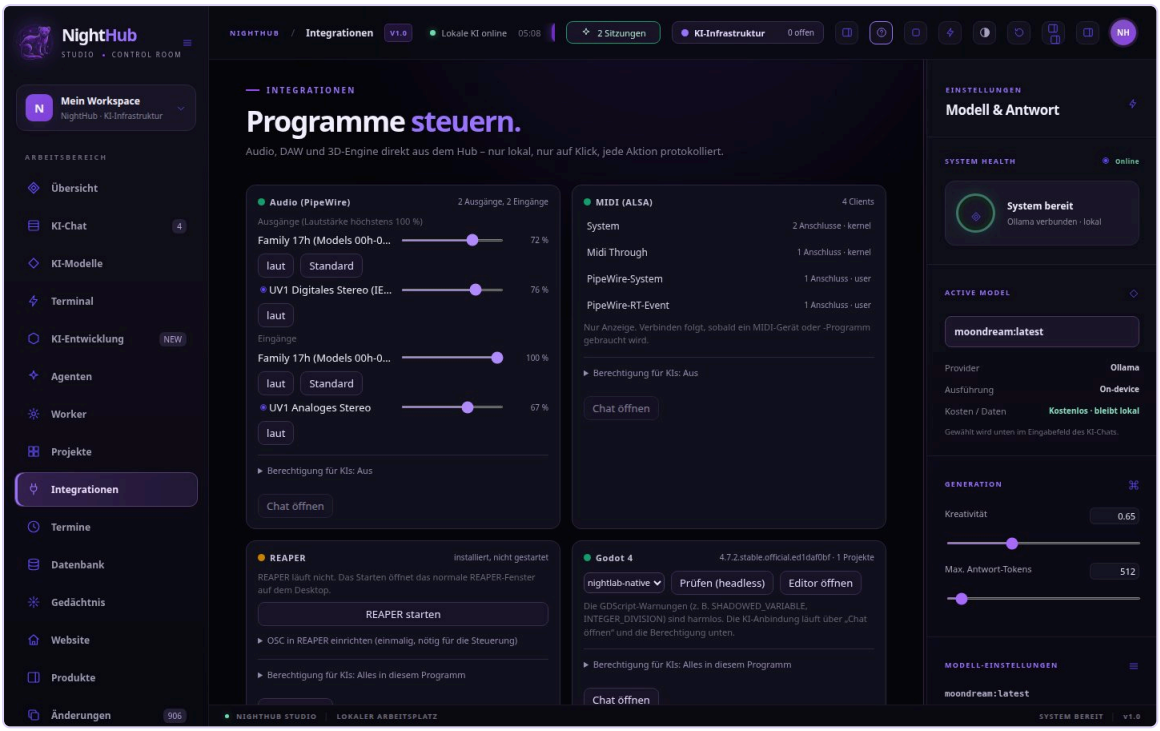
Live-Überblick über alle laufenden KI-Agenten: Claude-Code-Sitzungen, Codex, lokale Rollen und Notion-Agenten – wer arbeitet, wer wartet, in welchem Ordner. Neue Agenten lassen sich hier anlegen und starten.

06 Projekte



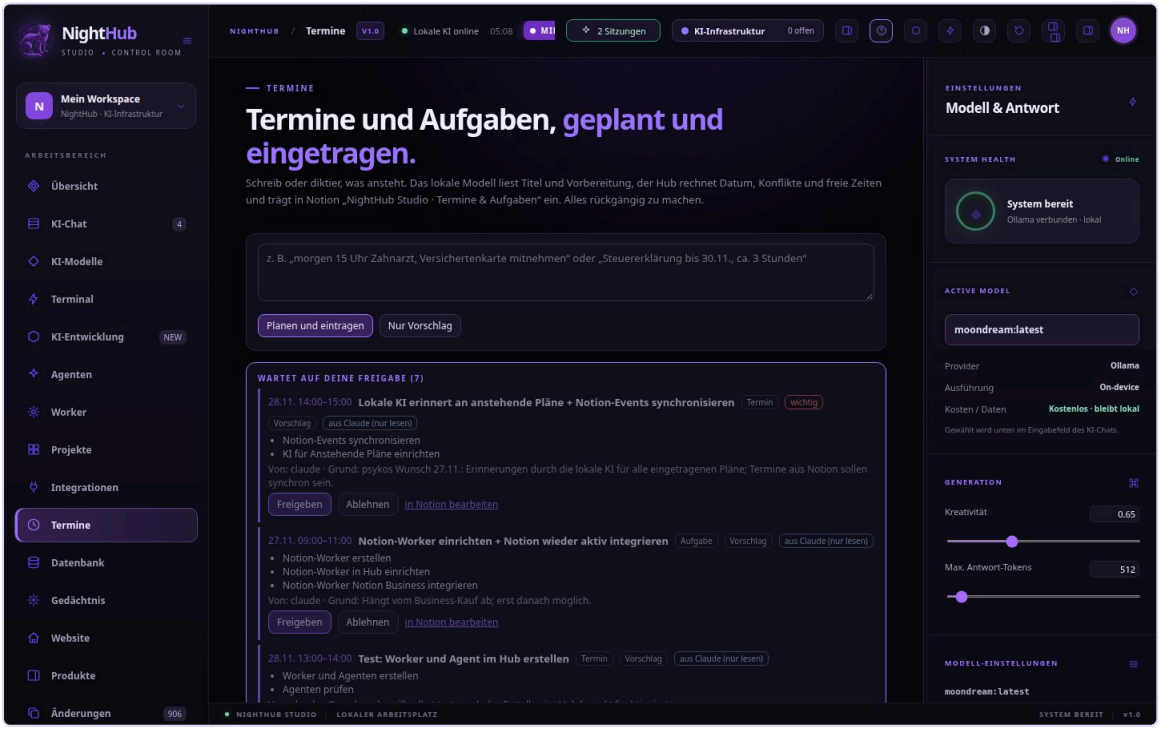
Ein kleines Büro: je Projekt Aufgaben, Termine, verknüpfte Ordner und der KI-Verbrauch der letzten sieben Tage. Gleiche Namen wie in Notion; alles liegt lokal.

07 Integrationen



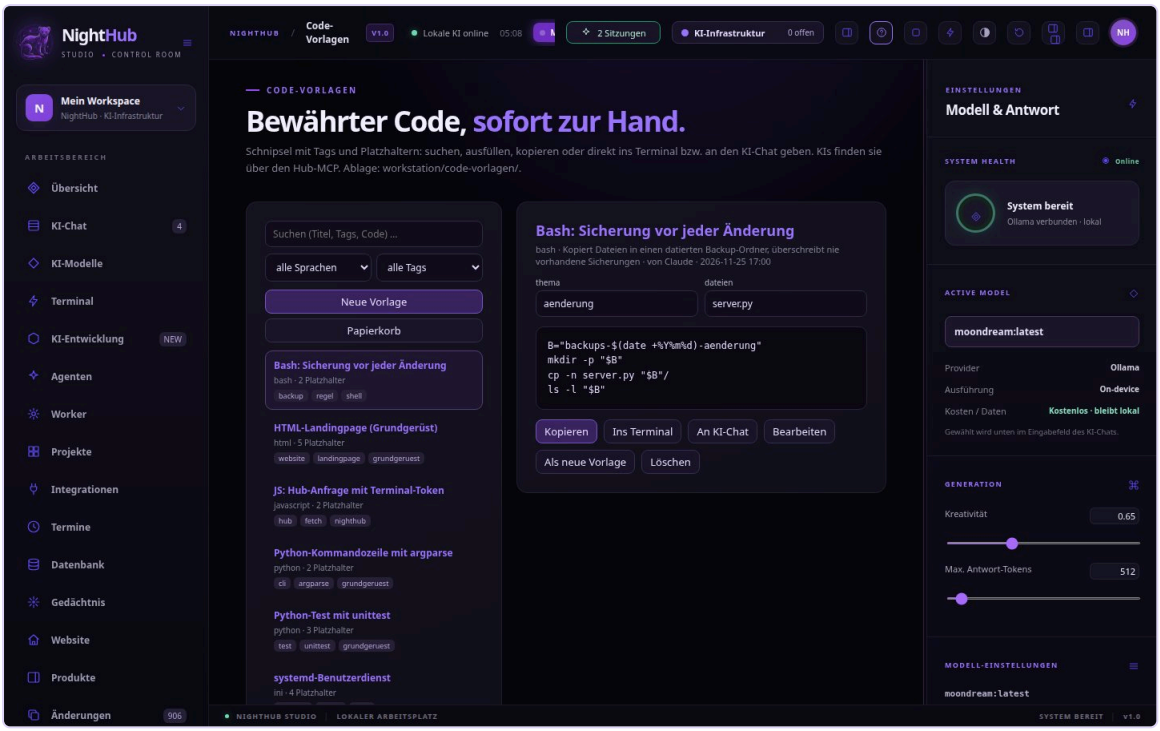
Programme direkt aus dem Hub steuern: Audio (PipeWire), MIDI, REAPER, Godot und weitere. Jede KI bekommt Rechte je Programm; jede Aktion wird protokolliert. Dazu ein eigener Chat, der Werkzeuge der Programme nutzen darf.

08 Termine



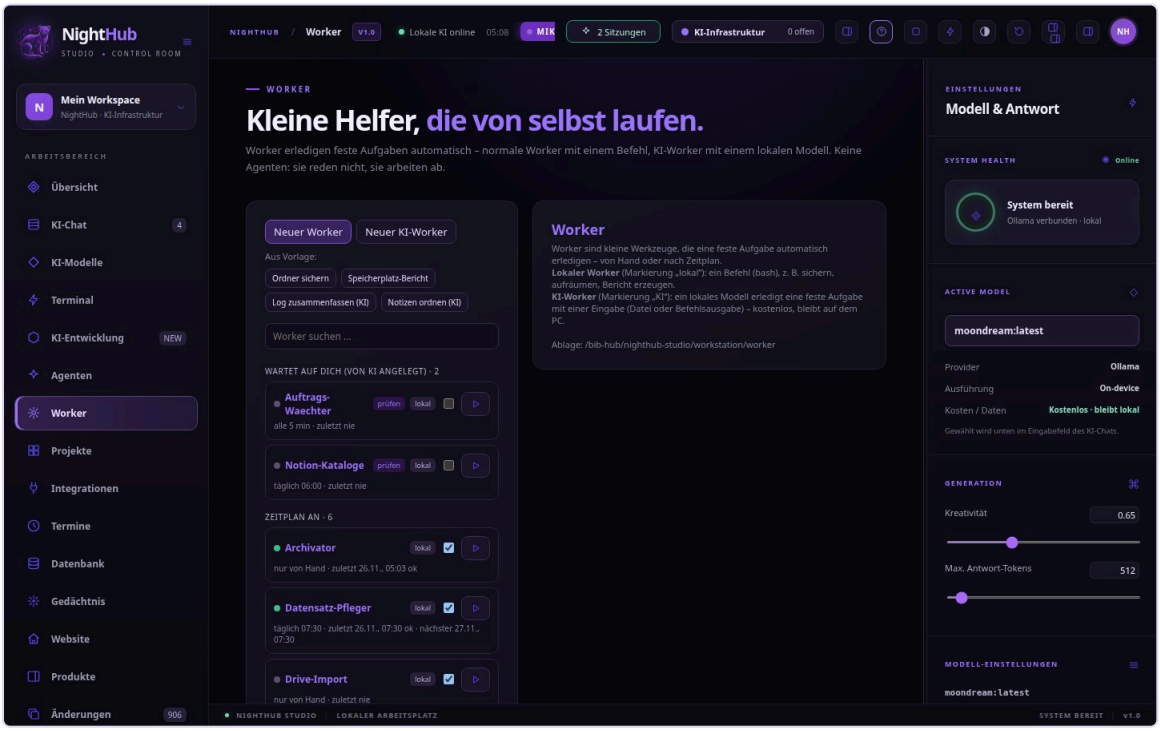
Termine in normaler Sprache schreiben oder diktieren. Ein lokales Modell liest Titel und Vorbereitung heraus, der Hub rechnet Datum, Konflikte und freie Zeiten. Vorschläge aller KIs warten auf Freigabe, bevor sie in Notion landen.

09 Code-Vorlagen



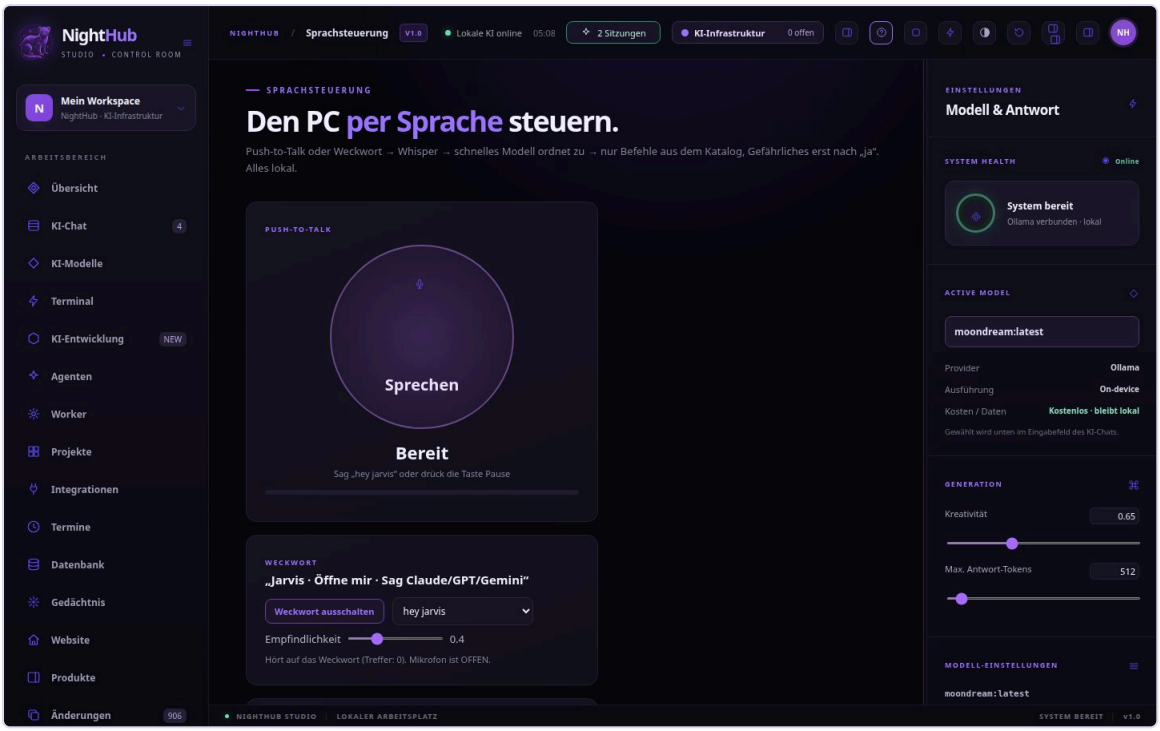
Bewährte Code-Schnipsel mit Platzhaltern und Tags: suchen, ausfüllen, kopieren oder direkt ins Terminal bzw. in den KI-Chat geben. Andere KIs finden und speichern Vorlagen über den Hub-MCP.

10 Worker



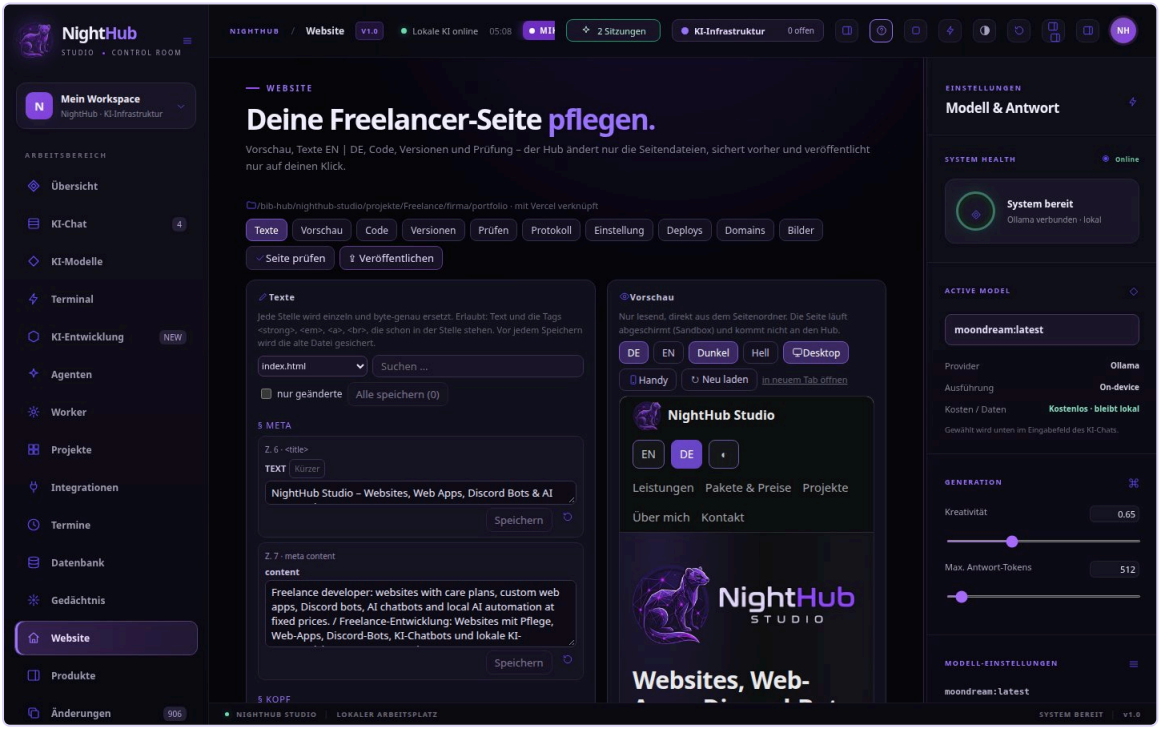
Kleine Helfer für feste Aufgaben: lokal (ein Befehl) oder mit einem lokalen Modell. Zeitplan, Protokoll, Ausgabedateien, Stoppen auf Klick. Von einer KI angelegte Worker bleiben aus, bis sie freigegeben werden.

11 Sprachsteuerung



Den PC per Sprache steuern: Push-to-Talk oder Weckwort, Whisper auf der Grafikkarte, ein schnelles Modell ordnet den Befehl zu – nur Befehle aus einem festen Katalog, Gefährliches erst nach einem gesprochenen „ja“. Alles lokal.

12 Website-Pflege



Die eigene Freelancer-Seite pflegen: Texte DE/EN einzeln bearbeiten, Vorschau hell/dunkel und Handy, Code, Versionen, Prüfung und Veröffentlichen. Der Hub ändert nur die Seitendateien, sichert vorher und veröffentlicht nur auf Klick.